Loops

- A **for loop** is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string).
- With the for loop we can execute a set of statements, for instance once for each item in a list or a string:

```
my_list = ['hello', 1, 3.0]
for item in my_list:
  print (item)
```

```
my_string = 'ATGTCGTATGC'
for item in my_string:
  print (item)
```

Else and nested loops

- The **else keyword** in a for loop specifies a block of code to be executed when the loop is finished:

else keyword

```
my_string = 'ATGTCGTATGC'
for item in my_string:
  print (item)
else:
  print ('done')
```

- A **nested loop** is a loop inside a loop.

Nested loop

```
my_string = 'ATGTCGTATGC'
my_list = ['hello', 1, 3.0]
for string_item in my_string:
  print (string_item)
  for list_item in my_list:
    print (list_item)
```

Loops – Try it out

```
numbers = ["one", "two", "three"]
for x in numbers:
  if x == "one":
    print(x)
  else:
    print ('Bazinga')
```

```
[27] # We will loop over a sequence and print out the position.
     # We keep track of the position whith the variable count.
     sequence = "CACTGACAC"
     count = 0
     for nucleotide in sequence:
       if nucleotide == "A":
         print('Nucleotide: ' + str(count), nucleotide)
       elif nucleotide == "T":
         print('Nucleotide: ' + str(count), nucleotide)
       elif nucleotide == "G":
         print('Nucleotide: ' + str(count), nucleotide)
       elif nucleotide == "C":
         print('Nucleotide: ' + str(count), nucleotide)
       count = count + 1
```

```
[28] # The range() function returns a sequence of numbers
     for x in range(6):
       print(x)
```

python™ https://colab.research.google.com/